22. (New) The method of claim 1, wherein the sum consists of the tagged numeric reference and the second tagged machine pointer.

23. (New) The method of claim 5, wherein:

the memory is subdivided into a plurality of pages;

the first object is stored on a first page; and

the second object is stored on a second page, other than the first page.

24. (New) The computer-readable medium of claim 11, wherein:

tag potions of the tagged numeric reference comprise N bits and store at least a first tag value and a second value indicating complementary properties; and

a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^{N}.

25. (New) The method of claim 1, wherein the sum consists of the tagged numeric reference and the second tagged machine pointer.

26. (New) The computer-readable medium of claim 15, wherein:

the memory is subdivided into a plurality of pages;

the first object is stored on a first page; and

the second object is stored on a second page, other than the first page.

REMARKS

By this amendment, claims 1-28 are pending, in which claims 21-28 are newly presented, and claims 3-7, 9-10, 13-17, and 19-20 are amended. No new matter is introduced.

The Office Action mailed October 24, 2002 rejected claims 5 and 15 under 35 U.S.C. § 102 as anticipated by *Lee.* (US 6,345,276), claims 1-2, 6-8, 11-12, and 16-18 as obvious under 35 U.S.C. § 103 based on *Lee* in view of *Murray* (Kelly E. Murray, "Under the Hood," Journal of Object-Oriented Programming, Sept. 1996, pp. 82-86), and claims 3, 9, 13, 19 over *Lee* and *Murray* further in view of *Carter et al.* (US 6,003,123).

Dependent claims 3-4, 6-10, 13-14, and 16-20 have been amended to improve the English and other matters of form, not to avoid the prior art. For example, claims 4, 10, 14, and 20 have been amended to provide a basis for the "N" in the formula "congruent to 2^{N-1} modulo 2^{N}."

The rejection of claims 5-9 and 15-19 is respectfully traversed because *Lee* alone or in combination with *Murray* fails to disclose the limitations of the claims. For example, independent claims 5 and 15 recite: "storing a reference within a first object to a second object in the memory as a numeric reference that encodes a location of the second object **as an offset from an address of the first object in the memory**." This is not shown in *Lee*.

Rather, *Lee* discloses a smart pointer that "contains two based addresses which are offsets **relative to the start of a shared memory heap**" (Abstract, emphasis added), not the "address of the first object in the memory" as presently recited in independent claims 5 and 15. Specifically, with reference to FIG. 2, object **203** at offset 20 in the heap **201** contains a smart pointer **204** to object **207** at offset 300 in the heap **201**. Smart pointer **204** itself stores in destination-pointer portion **206** the offset 300 of the object **207** in the heap **201**, but not the offset of the object **207** from the address of object **203**, which would be 300–20 = 280.

The rejection cannot be salvaged by reading the recited "first object," not on the object **203**, but on the entire heap **201**, because claims 5 and 15 further recite that "the first object and the second object do not overlap each other" (see, e.g. FIG. 2 of the present application for

adequate support) but the heap of *Lee*, however, **201** does overlap both object **203** and object **207**.

In addition, newly presented claims 23 and 26, which depend from claims 5 and 15, respectively, further recite that the memory is subdivided into pages and the first and second objects are stored on different pages, which are not detailed by *Lee*.

The rejection of claims 1-3 and 11-13 is respectfully traversed because modifying *Lee* to use the pointer tags of *Murray* would not result either in the claimed invention or in an operable embodiment. *Murray* discloses a Lisp implementation in which the bottom two bits of a machine word indicate the type of the object (p. 82, col. 3). Page 82, col. 1, show an assignment of tag values, in which 00 is for an integer, 01 is for a list point, 10 is for an other pointer, and 11 is nil.

Use of *Murray*'s tag values do not work in the *Lee* system, because the address calculation of *Lee* if combined with *Murray* would produce a pointer to a destination object that are tagged with the type of the source object. In other words, the pointer to would be tagged with the type of the wrong object. For example, in *Lee*, the address of object **207** is calculated in two steps, **317** and **319**. In step **317**, a base pointer is calculated by subtracting offset **205** from pointer **203**. Since offset **205** is a number, its tag according to *Murray* would be 00, and the base pointer would therefore have the same tag as pointer **203**. In step **319**, the base pointer is added to the destination offset **206** to produce the what should be, but is not, the tagged pointer to object **207**. Since the destination offset **206** is also a number, its tag is 00 and the result of step **319** is a pointer with the same tag as the base pointer, which has the same tag as the pointer to object **203**. However, object **207** can have a different type from that of object **203**, so the object **203**'s tag is wrong with respect to object **207**'s type. In fact, there is no assignment of type tags

in *Murray* that would result in the generated pointer to object **207** always having the right type tag.

Accordingly, combining *Lee* with *Murray* results in an inoperable system in which all destination pointers are tagged with the type of the source pointers. In order to fix the tag destination pointer, the source type tag must be masked off and the correct destination type tag must be logically or'd in. As a result, a functioning combination of *Lee* and *Murray* would no longer have a first tagged machine pointer that is a **sum** of a tagged numeric reference and a second tagged machine pointer. Different operations, such as masking, would have to be employed.

In addition, newly presented claims 21 and 24, which depend from claims 1 and 11, respectively, further recite a way of assigning complementary tag values such that a "a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^{N}." This is not shown in the *Murray* pointer tagging scheme.

Newly presented claims 22 and 25, also dependent on claims 1 and 11 respectively, further recite that the "sum **consists of** the tagged numeric reference and the second tagged machine pointer." This is not shown in *Lee* since *Lee*'s pointer arithmetic includes three terms: the address of object **203**, the negation of offset **204**, and the offset **205**.

The remaining dependent claims 2-3, 6-8, 12-13, and 16-18 are allowable for at the same reasons as their independent claims and are individually patentable on their own merits. For example, even though *Carter et al.* discloses a virtual memory system, there is no value in a tag portion that "indicates whether the first object has a same or a different contiguity as a contiguity of the second object" as recited in claims 3, 8, 13, and 18. Rather, the *Carter et al.* at best shows two different virtual page identifiers, in two different Global Translation Lookaside Buffer (GTLB) entries, which only indicate the number of the virtual page (cols. 17:62-67 and 18:15-

23). Merely knowing the virtual page numbers is not enough to know whether the first or second object is contiguous or discontiguous or whether their contiguity is the same or different.

Therefore, the present application, as amended, overcomes the objections and rejections of record and is in condition for allowance. Favorable consideration is respectfully requested. If any unresolved issues remain, it is respectfully requested that the Examiner telephone the undersigned attorney at 703-425-8516 so that such issues may be resolved as expeditiously as possible.

Respectfully Submitted,

DITTHAVONG & CARLSON, P.C.

6/24/2003
Date

Stephen C. Carlson
Attorney/Agent for Applicant(s)
Reg. No. 39929

10507 Braddock Rd
Suite A
Fairfax, VA 22032
Tel. 703-425-8516
Fax. 703-425-8518

APPENDIX

3. (Once Amended) The method of claim 1, wherein the tagged numeric reference includes a tag portion that indicates whether the first object [and the second object have] has a same or a different contiguity as a contiguity of the second object.

4. (Once Amended) The method of claim 3, wherein:

the tag portion includes N bits of the first tagged numeric reference that are less significant than bits used for an offset portion; and

the tag portion contains one of at least a first tag value indicating that the first object is contiguous and a second tag value indicating that the second object is non-contiguous, wherein a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^{N}.

5. (Twice Amended) A method of managing memory, comprising the computer-implemented steps of:

storing [a plurality of objects] a first object and a second object in a memory, wherein the first object and the second object do not overlap each other; and

storing a reference[s between the objects] within a first object to a second object in the memory as a numeric reference[s] that encodes a location[s] of [referenced] the second object[s] as an offset[s] from an address of the first object[s that reference the referenced objects] in the memory.

6. (Once Amended) The method of claim 5, further comprising the step of calculating a pointer difference between a first machine pointer to [a] the first object and a second machine pointer to [a] the second object to produce [a self-relative] the numeric reference.

7. (Once Amended) The method of claim 5, wherein [the step of calculating a pointer difference between a first machine pointer to a first object and a second machine pointer to a second object to produce a self-relative numeric reference includes the step of calculating the pointer difference between a first tagged machine pointer to the first object and a second tagged machine pointer to the second object to produce a tagged self-relative numeric reference] the first machine pointer is a first tagged machine pointer, the second machine pointer is a second tagged machine pointer, and the numeric reference is a tagged numeric reference.

9. (Once Amended) The method of claim 7, wherein a tag portion of the [self-relative] tagged numeric reference indicates whether the first object [and the second object have] has a same or a different contiguity as a contiguity of the second object.

10. (Once Amended) The method of claim 9, wherein:

the tag portion includes N bits of the tagged [self-relative] numeric reference that are less significant than bits used for an offset portion of the tagged numeric reference; and

the tag portion contains one of at least a first tag value indicating that the first object is contiguous and a second tag value indicating that the second object is non-contiguous, wherein a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^N.

13. (Once Amended) The computer-readable medium of claim 11, wherein the tagged numeric reference includes a tag portion that indicates whether the first object [and the second object have] has a same or a different contiguity as a contiguity of the second object.

14. (Twice Amended) The computer-readable medium of claim 13, wherein:

the tag portion includes N bits of the first tagged numeric reference that are less significant than bits used for an offset portion; and

the tag portion contains one of at least a first tag value indicating that the first object is contiguous and a second tag value indicating that the second object is non-contiguous, wherein a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^{N}.

15. (Twice Amended) A computer-readable medium bearing instructions for managing memory, said instructions arranged, when executed, to cause one or more processors to perform the steps of:

storing [a plurality of objects] a first object and a second object in a memory, wherein the first object and the second object do not overlap each other; and

storing a reference[s between the objects] within a first object to a second object in the memory as a numeric reference[s] that encodes a location[s] of [referenced] the second object[s] as an offset[s] from an address of the first object[s that reference the referenced objects] in the memory.

16. (Once Amended) The computer-readable medium of claim 15, said instructions further arranged to cause said one or more processors to perform the step of calculating a pointer difference between a first machine pointer to [a] the first object and a second machine pointer to [a] the second object to produce [a self-relative] the numeric reference.

17. (Once Amended) The computer-readable medium of claim 15, [the step of calculating a pointer difference between a first machine pointer to a first object and a second machine pointer to a second object to produce a self-relative numeric reference includes the step of calculating the pointer difference between a first tagged machine pointer to the first object and a second tagged machine pointer to the second object to produce a tagged self-relative numeric reference] the first machine pointer is a first tagged machine pointer, the second machine pointer is a second tagged machine pointer, and the numeric reference is a tagged numeric reference.

19. (Once Amended) The computer-readable medium of claim 17, wherein a tag portion of the [self-relative] tagged numeric reference indicates whether the first object [and the second object have] has a same or a different contiguity as a contiguity of the second object.

20. (Twice Amended) The computer-readable medium of claim 19, wherein:

the tag portion includes N bits of the tagged self-relative numeric reference that are less significant than bits used for an offset portion; and

the tag portion contains one of at least a first tag value indicating that the first object is contiguous and a second tag value indicating that the second object is non-contiguous, wherein a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^{N}.

21. (New) The method of claim 1, wherein:

tag potions of the tagged numeric reference comprise N bits and store at least a first tag value and a second value indicating complementary properties; and

a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^N.

22. (New) The method of claim 1, wherein the sum consists of the tagged numeric reference and the second tagged machine pointer.

23. (New) The method of claim 5, wherein:

the memory is subdivided into a plurality of pages;

the first object is stored on a first page; and

the second object is stored on a second page, other than the first page.

24. (New) The computer-readable medium of claim 11, wherein:

tag potions of the tagged numeric reference comprise N bits and store at least a first tag value and a second value indicating complementary properties; and

a difference of the first tag value and the second tag value is congruent to 2^{N-1} modulo 2^N.

25. (New) The method of claim 1, wherein the sum consists of the tagged numeric reference and the second tagged machine pointer.

26. (New) The computer-readable medium of claim 15, wherein:

the memory is subdivided into a plurality of pages;

the first object is stored on a first page; and

the second object is stored on a second page, other than the first page.